Attorney Docket: 003242.P015 *PATENT*

UNITED STATES PATENT APPLICATION

FOR

DIAGNOSTIC SCHEME FOR PROGRAMMABLE LOGIC IN A CONFIGURABLE SYSTEM ON A CHIP

First Named Inventor:

JEAN-DIDIER ALLEGRUCCI

PREPARED BY:

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
12400 WILSHIRE BOULEVARD
SEVENTH FLOOR
LOS ANGELES, CA 90025-1026

(408) 720-8300

"Express Mail" mailing label number EL672752457US

Date of Deposit: 11/3/00

I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to the Commissioner of Patents and Trademarks, Washington, D.C. 20231

JUANITA BRUCE
(Typed or printed name of person mailing paper or fee)

(Signature of person mailing paper or fee) Date

DIAGNOSTIC SCHEME FOR PROGRAMMABLE LOGIC IN A SYSTEM ON A CHIP

FIELD OF THE INVENTION

The present invention relates generally to diagnostic methods for programmable logic, and more specifically to accessing programmable logic in a system on a chip bus-based system.

BACKGROUND

A recent development in micro-electronics is the configurable system on a chip (CSOC). The system integrates a CPU, an internal system bus, and programmable logic, also referred to as configurable system logic (CSL). The various system resources are all interconnected, and communicating through an internal system bus, on a single piece of silicon. The internal system bus signals and various dedicated system resource signals that connect to the CSL are collectively referred to as the configurable system interconnect or CSI. There are two types of pins, dedicated pins to interface with external devices (e.g., external memory) and programmable pins that can serve as an interface to other user logic. The dedicated processor bus and system resources provide an efficient and stable high performance system, while the configurable system logic provides flexibility for the user to implement additional functions. There are many benefits to embedding the programmable logic, including time-to-market, integration, and flexibility. The downside of embedding the programmable logic is that the signals are not directly accessible (i.e., observable and controllable) by the engineer charged with system debugging. Many of the signals that are of considerable interest when debugging a system are now buried inside the device. As a result, system debugging and trouble-shooting capability can be severely limited.

SUMMARY OF THE INVENTION

A method is described for diagnosing programmable hardware in a programmable logic system. The method comprises ceasing bus access upon the occurrence of a specified event or sequence of events while allowing the completion of all pending transactions. When all pending transactions are completed the system clock is stopped such that the state of the programmable hardware is held static. The static state of the hardware is then accessed through a debug port. An apparatus and a machine readable medium that implement the method are also described.

Other features and advantages of the present invention will be apparent from the accompanying drawings and from the detailed description that follows below.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limitation in the figures of the accompanying drawings, in which like references indicate similar elements and in which:

Figure 1 is a block diagram of a configurable system on a chip;

Figure 2 is a timing diagram for the clock freeze operation; and

Figure 3 is flow chart of the clock freezing process of an embodiment of the present invention.

DETAILED DESCRIPTION

An embodiment of the present invention will provide a more efficient method of debugging user-implemented hardware in a configurable system on a chip. An embodiment of the invention allows the user to freeze the clock upon the occurrence of a user-specified event while avoiding the possibility that the clock will be frozen in a wait state. In one embodiment of the present invention the breakpoint unit of the system is programmable and will issue a clock freeze event upon the occurrence of a programmed event or sequence of events. The bus arbiter will cease granting bus access at this time, but will allow all pending transactions to be completed. The system can be stopped and the state of the system at a particular point can be viewed for debugging purposes. This method provides the user with a "snapshot" of the system at a desired time.

An intended advantage of one embodiment of the invention is to provide the user with the state of the system at a given time for debugging purposes. Another intended advantage of one embodiment of the invention is to ensure that the system is not currently in a wait state when the system clock is stopped. This allows the bus to be used by the debugging port.

Figure 1 is a block diagram of a configurable system on a chip. The system 100 shown in Figure 1 includes those portions of a CSOC relevant to an embodiment of the present invention, although a variety of different computing systems can implement the present invention. The system 100 shown in Figure 1 includes a configurable system logic 105, which is coupled to the rest of the system through the configurable system interconnect 110 CSI. The rest of the system 125 includes the CPU, DMA, peripherals, counters, timers, memory, and memory interfaces. The system includes a debug joint test action group (JTAG) port 115 that is connected to

a user input device 120 that is external to the chip, for example, a computer. The debug JTAG port 115 is a busmaster on the CSI bus 110 and therefore has access to any resources connected to the CSI bus 110. This includes the breakpoint unit 130, so that the user can program the breakpoint unit 130 through the JTAG port 115. In one embodiment the breakpoint unit 130 connects to, and allows tracing of, multiple buses and includes the ability to break on the occurrence of a predetermined bus event on any one of the multiple buses. In one embodiment the breakpoint unit 130 may be connected to, and programmed by, a host debugging system via a port on the target chip.

The breakpoint unit monitors the CSI bus 110. The user programs the breakpoint unit 130 to break on a specific condition or sequence of events. The breakpoint unit 130 may be configured to generate one or more output signals upon a breakpoint event. The output signals may be used to interrupt of freeze a processor, depending on the processor's supported features. So, for example, the user may program the breakpoint unit 130 to break as soon as there is a write to a specific register address. As soon as that happens, the breakpoint unit 130 generates a breakpoint event (e.g., the breakpoint unit 130 generates a clock freeze cycle). The clock freeze signal is propagated to the system clock. However, due to the pipeline nature of the bus, there may be pending operations from the bus that are being executed. For example, before the write signal that triggered the breakpoint there may have been a read that was being executed. That read may be to an external memory device that takes several cycles to execute and may, therefore, be in a wait state. There could be many other examples of accessing something in the configurable system logic that might have generated a wait state prior to the breakpoint event. All of these transactions must be terminated prior to freezing the

clock. If the transactions were not completed the system might be frozen while in a wait state, which would render the CSI bus 110 inoperable. The bus arbiter 135 is monitoring the CSI bus 110. In one embodiment the bus arbiter 135 is a state machine that implements a round-robin arbitration algorithm. One function of the bus arbiter 135 is to receive access requests from the several bus masters and grant access to a particular bus master after each clock cycle based on the arbitration algorithm. Another function of the bus arbiter 135 is to keep track of all transactions on the CSI bus 110 for debugging purposes. When the bus arbiter 135 receives a clock freeze signal from the breakpoint unit 130, the bus arbiter 135 stops granting access on the CSI bus 110. The bus arbiter 135 then waits for pending transactions to be completed and then allows the clock to be frozen, because only at this point can it be guaranteed that there won't be any wait states generated and that's because there are no more transactions pending on the bus.

Figure 2 is an example of a timing diagram for the clock freeze operation discussed above in reference to Figure 1. At T_0 Figure 2 is an example of a waveform diagram for the clock freeze operation discussed above in reference to Figure 1. The system clock 205 is functioning, the CSL clock 220 is functioning and therefore any requests, for example a DMA bus request 225, is granted. The DMA grant signal 230 is high, indicating that requests, for example DMA request 225, are being granted. At some time, T1, a clock freeze event occurs and the clock freeze event 210 goes high. At this point the arbiter stops granting any requests. The DMA grant line 230 goes low, indicating that no CSI requests will be granted. However, the qualified clock freeze signal 215 remains low, and remains low until some time, T3, where the last pending transaction is completed. The bus arbiter is aware of the last pending transaction completion. When the last pending transaction is complete

the bus arbiter transmits the qualified clock freeze signal, and qualified clock freeze 215 goes high. The qualified clock freeze signal freezes the CSL clock as indicated by CSL clock signal 220 which stays high during the debugging process.

Figure 3 describes the process by which the clock is frozen accordance with one embodiment of the present invention. Process 300 shown in Figure 3 begins at operation 305 in which a breakpoint event occurs. The breakpoint unit has been programmed by the user to break on specific conditions or sequences of events. At operation 310, the breakpoint unit sends a clock freeze signal to the bus arbiter. The bus arbiter then stops granting requests for the bus at operation 315. At operation 320, the bus arbiter checks for any pending operations on the bus. If there are pending operations on the bus, the bus arbiter checks to see if they are complete in operation 325. If they are not complete, the arbiter continues to monitor. If the arbiter finds that they are complete in operation 330, the arbiter then sends the qualified clock freeze signal to the CSL clock and the system is frozen. At this point, because there are no further transactions pending, there will be no wait state generated. Therefore the problem of freezing the system in a wait state is avoided. The user may now access the system from the bus and continue with debugging through the JTAG debugging port.

The process of the present invention may be implemented through use of a machine-readable medium that includes any mechanism that provides (i.e. stores and/or transmits information in a form readable by a machine (e.g., a computer). For example, a machine-readable medium includes read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; electrical, optical, acoustical or other form of propagated signals (e.g., carrier waves, infrared signals, digital signals, etc.); etc.

In the foregoing specification, the invention has been described with reference to specific exemplary embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.